Digital Content Store API Documentation

Digital Content Store

DCS Request and Content API

Technical specification

Version 3.0

Document Control

Change Control

Version	Name	Position	Date	Description
0.1	Kieran Burke	Business Analyst	04 Oct 2016	First draft
0.2	Alex Cole	Senior Business Analyst	11 Oct 2016	Review and changes
1.0	Alex Cole	Senior Business Analyst	19 Oct 2016	Final version
1.1	Kieran Burke	Business Analyst	07 Apr 2017	Removed description from '/GetCourseContent' call
2.0	Kieran Burke	Business Analyst	06 Mar 2018	Version 2 of the API
2.1	Steve Collingham	Senior Business Analyst	20 Apr 2018	Updated for V2 of the API
3.0	Ben Ridout	Application Support Manager	31 July 2019	Version 3 of the API. Addion of 2 new methods: /AmendCourse /SubmitCourse New parameter (in the /GetCourseContent call allowing a specific Request to be returned).

Table of Contents

Table of Contents

Introduction

Purpose

This document describes the Digital Content Store (DCS) Course Content API, which allows a client to get course data and content links and post new requests and new courses and to amend existing courses, where the institutions involved with such operations have allowed it. It is intended that this service will be used by Third Party Vendors of technology to CLA Higher Education Institutions (HEIs) or by CLA Licensed HEIs to serve content links in their systems.

Scope

This document includes the four methods currently implemented in the API, along with specifications and examples. Changes that may occur in the API will provoke a new document being provided. This document will not include any information about how the data should be used or manipulated in a system, though sequence diagrams are provided to show call order scenarios.

The DCS API has three GET, two POST and one PUT method. Responses are always given as JSON data.

Glossary

Term	Meaning
DCS	The Digital Content Store, a web application provided to CLA licensees in the Higher Education Sector
API	Application Programming Interface is a source code based specification intended to be used as an interface by software components to communicate with each other
HTTP	Hypertext Transfer Protocol. An application protocol for exchanging files (text, graphic images, sound, video, and other multimedia files) on the Web
GUID	Globally Unique Identifier
JSON	JavaScript Object Notation
URL	Uniform Resource Locator. The address of a resource accessible on the Web. The URL includes the name of the protocol required to access the resource, identifies the address of a specific server on the Web, and contains a hierarchical description of a file location on the server
DOI	Digital Object Identifier
ISBN	International Standard Book Number. Uniquely identifies a single manifestation of a work. ISBNs are used for monographic publications
ISN	International Standard Number. A generic term referring to both ISSN and ISBN identifiers
ISSN	International Standard Serial Number. Uniquely identifies a journal. ISSNs are used for periodical publications
Manifestation	An individual book or serial. Different published editions of the same textual work are different manifestations. (For example, the Penguin Popular Classic edition of Pride and Prejudice is a separate manifestation to the Oxford World Classics edition)
HEI	Higher Education Institution
Course	A Course of Study: any whole course of study or any module or segment of a student's studies which is normally regarded by the Licensee as a discrete and self-contained unit for the purposes of examination or assessment or, in the case of a non-credit bearing course, that particular course
Content	A PDF item
OCR	Optical Character Recognition

Service Description

This section gives an overview of the Digital Content Store (DCS) Request and Content API and gives a high level description of the functionality. The subsections will include details on the methods in place, some information on the interaction of the DCS with the API, and important knowledge about system activity and periodic events.

Overview

The DCS is a workflow tool for librarians in HEIs to help with the process of sourcing and clearing digitisation requests required for course packs and reading lists. Documents are stored in the system as PDF files. When a request is completed a URL is published that allows authorised users at the institution to access the content. Each content URL is specified by course and requires the end user (typically students) to authenticate to access the content. The DCS API allows external systems to publish the content links and associated metadata on a course by course basis. CLA will create the client system credentials within the DCS so that administrators of the HEI accounts can let the clients subscribe to their courses.

The API is comprised of six methods; *GetInstitutions*, *GetCourses*, *GetCourseContent* and *SubmitRequest SubmitCourse*, *AmendCourse*. Content retrieval is only possible using the HEI and course codes provided by the relevant methods.

Getting Started

The following steps need to be completed

1. CLA sets up the API user account on the DCS Sandbox
2. DCS user subscribes the newly created API User to their organisation in the DCS Sandbox
3. HEI user sets up an account on the CLA API portal (https://apiportal.cla.co.uk/), and subscribes to the Digital Content Store Product.
4. CLA approves user on the API portal
5. Testing can commence on the demo environment
6. Once testing is complete, the HEI user should let CLA know that they are ready to use the Production API. CLA will then create an API User on the DCS production system.
7. DCS user subscribes the newly created API User to their organisation in the DCS production
8. The production environment can now be used

Architecture and versioning

The DCS API uses the HTTP protocol and returns data in the JSON format.

The URL format will be https://api.cla.co.uk/apiname/version/method

In the case that no version is supplied, v1 will be defaulted for backward compatibility reasons.

Subscription

Once the API account has been added to the DCS system, HEI users (HEI administrator level only) will have the option to subscribe the API User to their institution. Once they have allowed the subscription, their details and courses and content will be available to the calling API system.

As HEIs only appear in the *GetInstitutions* response when they have allowed the subscription to the API system, you will have to notify the HEI that your system is now available, as CLA do not have the right to subscribe on behalf of an HEI. HEIs can be referred to our helpdesk and knowledgebase available at https://cla.zendesk.com/hc/en-us.

Get Institutions

This service can be used to check which HEIs have enabled access to your platform/service and to retrieve the HEI IDs needed on the subsequent calls. If you need help subscribing to your own use of the API, or your customers do not know how to subscribe to your use of the API, please contact / have them contact support@cla.zendesk.com.

Get Courses

Using the retrieved HEI ID, this method will return a list of courses in the current academic year for the nominated HEI, along with the data relating to the course. Each course has a course code that can be used for identifying the course in the *GetCourseContent* calls. This call can also be used to check whether courses already appear in the nominated HEI's instance of the DCS and could be used as a precursor to the *SubmitCourse* call.

Get Course Content

The GetCourseContent call uses the institution id, and course code or request Id, to return relevant Course Content. Using Course Code will retrieve all requests for that course and using the request id will just the specific request. This call is used to return all relevant associated metadata from a request in the DCS, including the link to content. These results can then be displayed in your own system/s.

Submit Request

This method gives the user the ability to submit a request into the DCS, which they can track through the other available methods. This supports books and journals.

Submit Course

This method gives the user the ability to submit a new course into the DCS. We recommend using the *GetCourses* call first, to check whether the course is already available in the nominated HEI's instance of the DCS, before you try to submit a new course.

If the Number of Weeks parameter is not included in the call, or if it is given as '0' then the system will default to '52' weeks. If the Number of Students is not given in the call, then this will always default to '0'. If you do not wish for these to default, please ensure you include the Number of Weeks (a numerical value between 1 and 52) and/or the Number of Students (a numerical value between 0 and 9,999) in the call.

Amend Course

This method gives the user the ability to amend an existing course in the DCS. This includes archiving, reinstating and deleting courses. We recommend using the *GetCourses* call first, to check whether the course is already available in the nominated HEI's instance of the DCS, before you try to amend a course.

If the Number of Weeks parameter is not included in the call, or if it is given as '0' then the system will default to '52' weeks. If the Number of Students is not given in the call, then this will always default to '0'. If you do not wish for these to default, please ensure you include the Number of Weeks (a numerical value between 1 and 52) and/or the Number of Students (a numerical value between 0 and 9,999) in the call.

When updating the status, 'Archived' will make an 'Active' course 'Archived' (content will therefore no longer be accessible). 'Active' (the default for submitting new courses) will reinstate 'Archived' courses (making content accessible again). Delete will permanently delete a course (and all the content within) and no other status overrides will be possible. We suggest approaching delete with great caution (perhaps even having a warning message) or else we suggest avoiding the deleting update completely

State Changes in Courses and Content

See Appendix 2 for content status definitions.

Courses may be in one of the following states:

- Active
- Archived
- Deleted

All course statuses will be returned in the response.

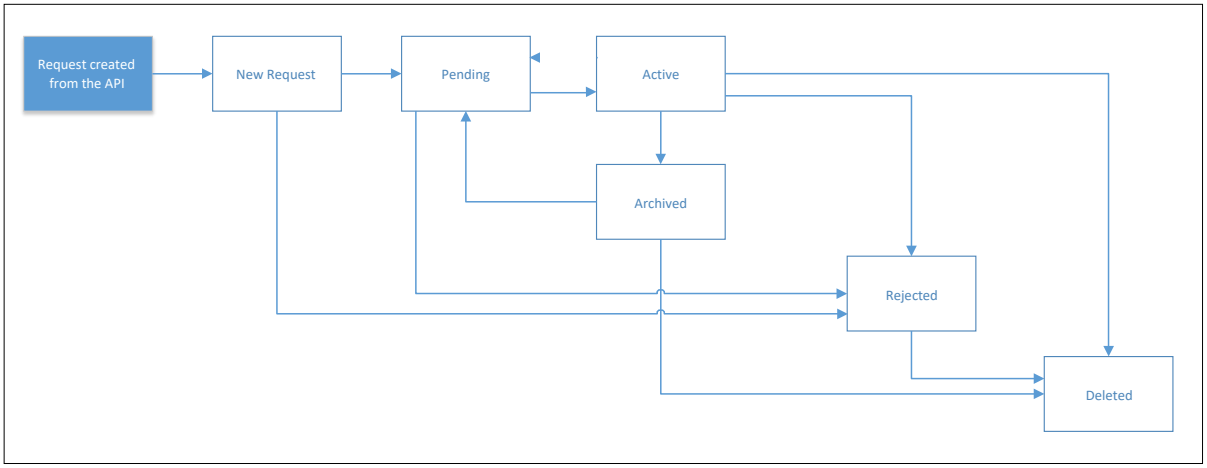
In the DCS, if a course is deleted, then the users are free to create a new course with the same course code as the deleted course. This means that there could be duplicate course codes in this response.

Content items in the DCS may be in one of the following states:

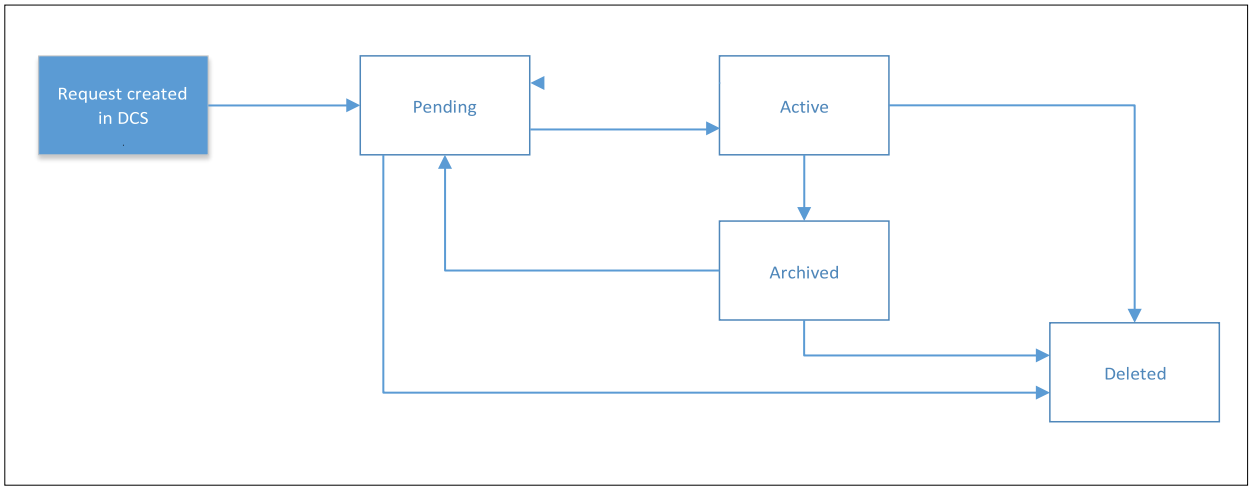
- New Request
- Rejected
- Active
- Archived
- Deleted
- Pending

HEIs are responsible for managing the changes of state. All states will be returned in the *GetCourseContent* call. This is to say, if a piece of content is moved from active to archived or pending and vice versa, then this change will be seen in the content-status field. Links to content are only provided for content items which have been in the active state at some point in time.

For a request made via the API (rather than directly through the DCS itself), the following states can be reached;



For a request made directly through the DCS, the following states can be reached;



The fundamental difference between the two, is that for a request made directly through the DCS, there is no ability to reject a request, it can only be deleted.

Starting a new academic year

See Appendix 2 for content status definitions.

See Appendix 3 for a chart explaining the timeline of events.

When an institution wants to move into the next academic year, their courses will also all be carried over, and this will be shown by the academic-year field in the *GetCourses* call. This is typically performed between 1st June and 15th June.. Since the *GetCourses* call can only be used to get the courses in the current academic year, once an institution has started the next academic year, courses in the previous academic year will no longer be able to be queried via the API.

As a result of the creation of a new academic year, within the DCS, and the carrying across of existing content, each existing request will be updated with a new request ID. The previous year's request ID can still be found in the field "previous-year-id", whereas the content will be given a new request ID "request-id" which relates to the new academic year.

For any external applications which are storing the "request-id" there will need to be an automated process to loop through all items retrieved through the API call "GetCourseContent" to search for any items which have been updated, and use the new "request-id" instead of the "previous-year-id". Between the 1st June and 15th June, API users may wish to use the field“previous-year-id” in their calls, as opposed to “request-id”, until their request-ids have been updated.

You can also search by Request ID as opposed to the Course Code using the *GetCourseContent* call, and this will be able to call back all the relevant up-to-date details for the request, even when using an old Request ID. Therefore, if using the Request ID to call back content, you can ignore the steps above during rollover. The Request ID is provided at the point a *SubmitRequest* call is placed into the DCS and, for requests that originated from the DCS, Request IDs are available in the *GetCourseContent* call when using Course Code as the search parameter.

In the first week of August, CLA will recheck all the items used under the CLA licence to verify that course content is covered in the following academic year under the CLA licence or permissions need to be sourced elsewhere. This may cause course content to go from Active to Pending if the permissions change. Librarians and academic support staff will be able to override these to be once again covered by the CLA Licence, or else source alternative permissions or content at this time, triggering changes in content status and availability of content.

Technical Specification

Introduction

Authentication

The DCS API uses a standard “basic” authentication scheme. All calls to the API need to include an HTTP header with appropriate credentials. The authorization field consists of a username and a password combined with a single colon encoded in base64 prepended by the word “Basic”. The HTTP request’s *Authorization* header should then be set with the resulting string.

The process to create a proper authentication header is as follows:

1. Define a string in the following format: “username:password” (ignore quotes).
2. Convert this string into a base64-encoded string.
3. Prefix the word “Basic” (again, ignore quotes) to the base64-encoded string, separated by a space.
4. Set the HTTP request’s authorization header with the final string.

For example, if the caller has the username “Foo” and the password “Bar” the resulting header would look like:

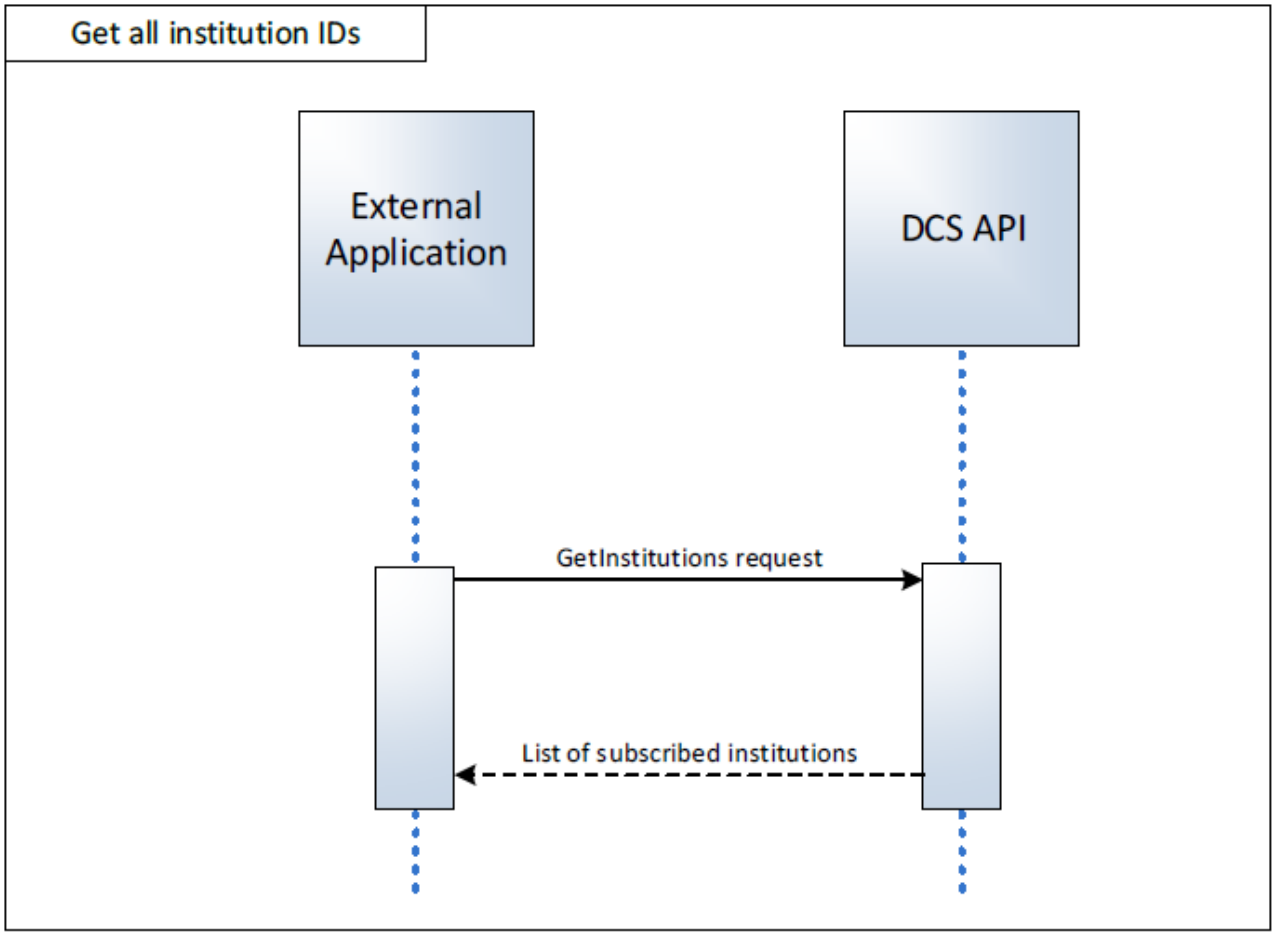
```
authorization: Basic Rm9vOkJhcg==
```

In order to ensure the security of this scheme, all calls will be over SSL using a recognized security certificate, thus resulting in only encrypted data sent to and from the DCS Course Content URL API. More information on basic authentication can be found here (RFC2617): https://tools.ietf.org/html/rfc2617#section-2

Workflow and sequence diagrams

Get all institution IDs

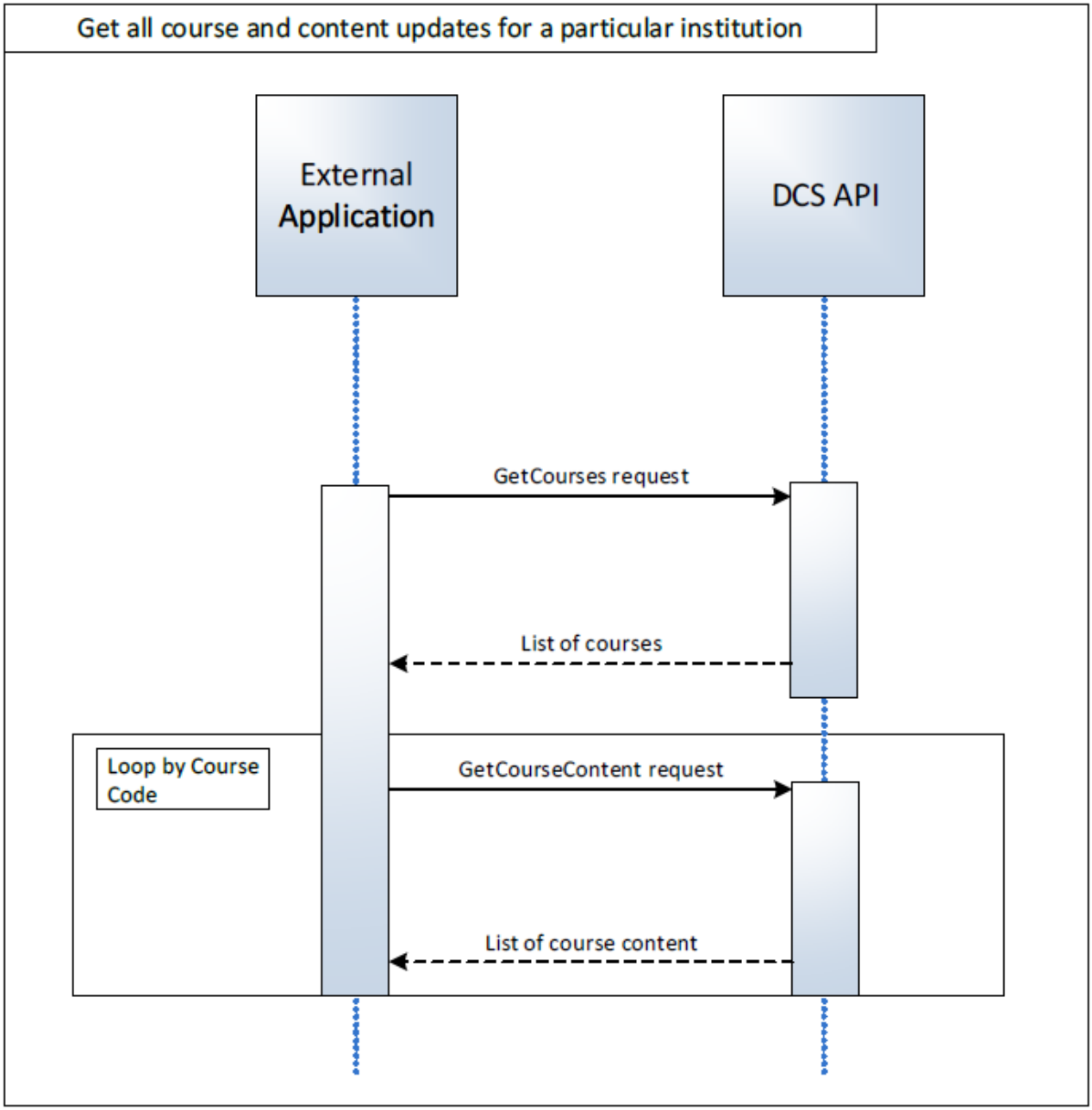
This sequence will retrieve the IDs that are associated with the HEIs that have allowed subscriptions. This can be used to confirm that an institution has granted access.



Get all course and content updates for a particular institution

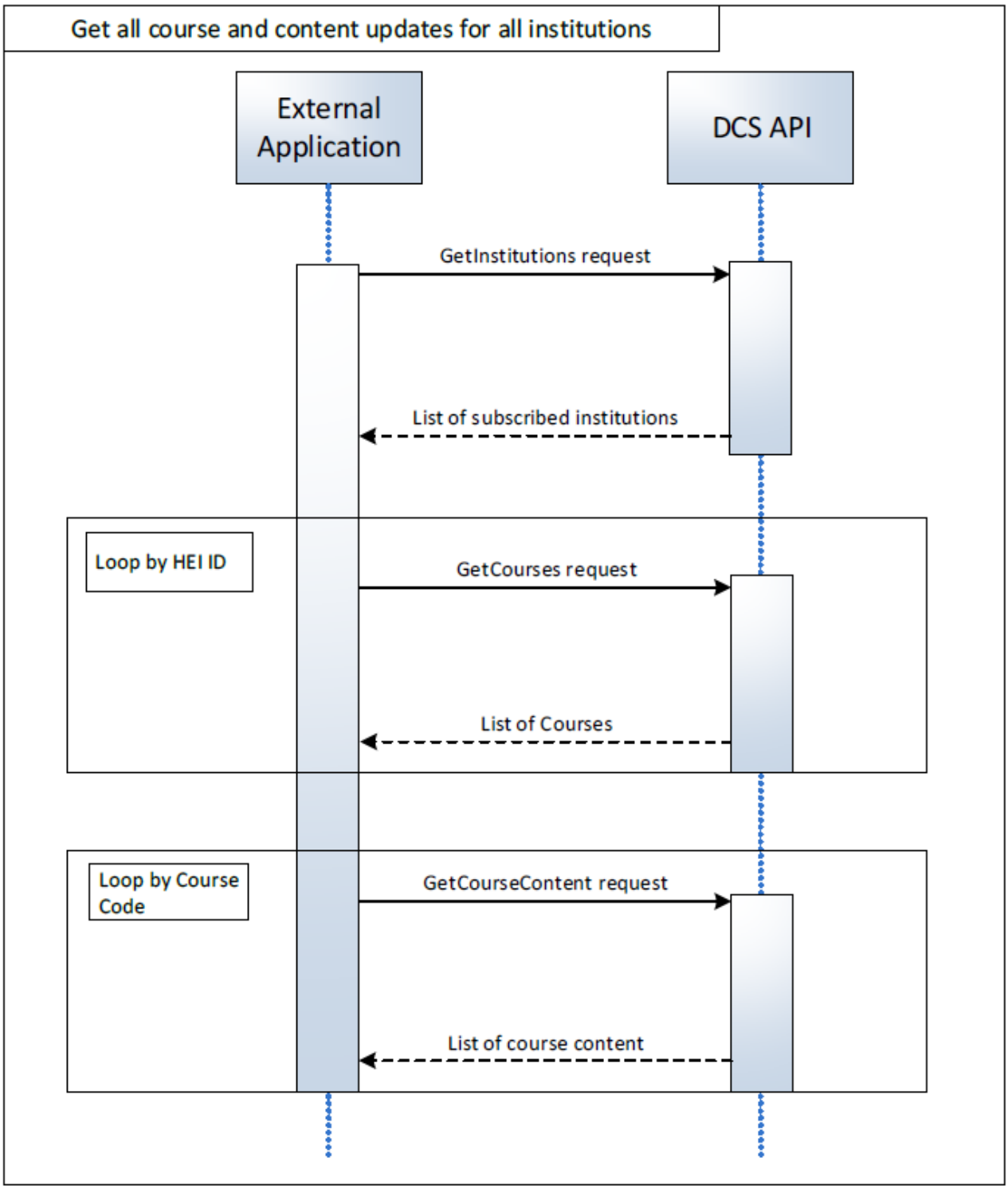
For a specified institution obtain the list of courses and the course content. Also used to check for new courses, updates to courses and updates to content. To use this process, the institution must already know their HEI ID, which is included in the call to GetCourses and GetCourseContent. If the HEI ID is not known, it can be retrieved by first calling the GetInstitutions method, as per the next sequence described (Get all course and content updates for all institutions).

This sequence can be used to update a specific institution courses and content items.



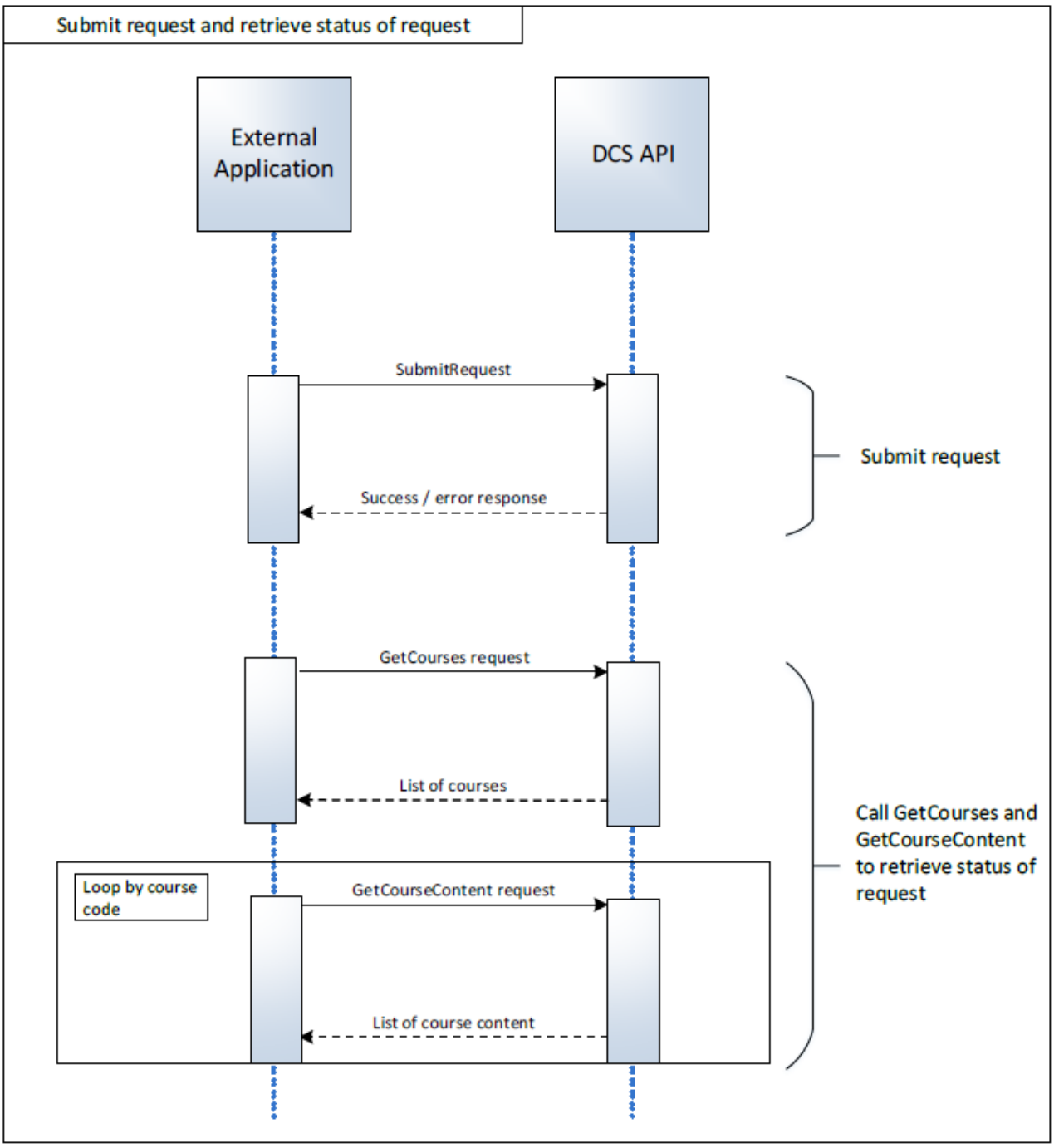
Get all course and content updates for all institutions

This sequence can be used to provide the full set of data so the client system can be completely updated to match the current status of the Digital Content Store. This is to be used at times when all institutions need to be updated, such as after CLA annual permissions recheck.



Submit request and retrieve status of request

This sequence can be used to submit a request for content to the DCS and then retrieve the status for the request later.



Get Institutions

URL: /GetInstitutions

Method: GET

Authentication: Required

Parameters: None

Success Response

Field	Type	Description
status	string	The status of the API response ("ok", "error")
status-code	int	Status-code of the API response
status-message	string	Message related to the API status code (if applicable)
total-results	int	Number of Institutions returned
Institutions	object[]	List of institutions associated to the API user
id	int	Institution ID
name	string	Institution Name

```
{
  "status": "string",
  "status-code": 0,
  "status-message": "string",
  "total-results": 0,
  "institutions": {
    "id": 0,
    "name": "string"
  }
}
```

Error Response

```
{
   "status": "error",
   "status-code": 3,
   "status-message": "Could not authenticate user"
}
```

Sample Call

```
 GET https://{host}/GetInstitutions HTTP/1.1
authorization: Basic YourBase64encodedusername:password
```

Sample Response

GetInstitutions Response

```
{
        'institutions':
        [
                {'id': 209, 'name': 'API_TEST'},
                {'id': 210, 'name': 'API_TEST_OTHER'}
        ],
        'status': 'ok',
        'status-code': 100,
        'status-message': 'Success',
        'total-results': 2
}
```

Get Courses

URL: /GetCourses?hei={hei}

Method: GET

Authentication: Required

Parameters**:**

Parameter name	Type	Value type	Description	Mandatory
hei	Query	integer	The ID of the HEI being queried	Yes

Success Response:

Field	Type	Description
status	string	The status of the API response ("ok", "error")
status-code	int	Status-code of the API response
status-message	string	Message related to the API status code (if applicable)
total-results	int	Number of Courses returned
courses	object[]	List of Courses associated to the API user
-academic year	string	Academic year the course is currently in
-id	int	Id of the current course

course-code	string	The course code of the current course
name	string	Name of the course
duration	int	Duration of the course in weeks
lecturer	string	The lead lecturer assigned to the course
status	string	The status of the course ("Active", "Archived", "Deleted")


```
{
  "status": "string",
  "status-code": 0,
  "status-message": "string",
  "total-results": 0,
  "courses": {
    "academic-year": 0,
        "status": "string",
    "id": 0,
    "course-code": "string",
    "name": "string",
    "duration": 0,
    "lecturer": "string"
  }
}
```


Error Response


```
{
   "status": "error",
   "status-code": 1,
   "status-message": "Institution not found"
}
{
   "status": "error",
   "status-code": 5,
   "status-message": "User not subscribed to HEI"
}
{
   "status": "error",
   "status-code": 6,
   "status-message": "Invalid Parameter"
}
```


Sample Call

```
GET https://{host}/v3/GetCourses?hei={hei}

authorization: Basic YourBase64encodedusername:password
Ocp-Apim-Subscription-Key: {subscription key}
```

Sample Response

GetCourses Response


```
{
    'courses':
    [
        {
        'academic-year': '2016-2017',
               'status': 'Active',
        'course-code': 'HIST101',
        'duration': 52,
        'id': 70668,
        'lecturer': 'Jane Bunt',
        'name': 'Introduction to World History'
        },
        {
        'academic-year': '2016-2017',
               'status': 'Archived',
        'course-code': 'ENG101',
        'duration': 26,
        'id': 70918,
        'lecturer': 'Steve McGill',
        'name': 'Introduction to English Language'
        },
        {
        'academic-year': '2016-2017',
               'status': 'Deleted',
        'course-code': 'LIT500',
        'duration': 52,
        'id': 70919,
        'lecturer': '',
        'name': 'English literature and the works of Shakespeare'
        }
    ],
    'status': 'ok',
    'status-code': 100,
    'status-message': 'Success',
    'total-results': 3
}
```

Get Course Content

URL: /GetCourseContent?hei={hei}&code={code}

Method: GET

Authentication: Required

Parameters:

Parameter name	Type	Value type	Description	Mandatory
hei	Query	integer	The ID of the HEI being queried	True
code	Query	string	The course code of the course being queried	True if ‘Request-ID’ is null
Request-ID	Query	integer	The Content Request Id being queried	True if ‘code’ null

Success Response

Field	Type	Description
status	string	The status of the API response ("ok", "error")
status-code	int	Status-code of the API response
status-message	string	Message related to the API status code (if applicable)
total-results	int	Number of Requests returned
HEI	string	Name of the HEI
course-ID	int	ID of the course the data comes from
content-items	object[]	List of content items associated to the given course
- content-GUID	guid	Unique Id for the content link
- content-URL	string	Content item’s current DCS link
- request-id	integer	Request's internal ID for the current academic year
- previous-year-id	integer	Request's internal ID for the previous academic year
- content-status	string	Content’s status, possible outputs: • New Request • Rejected • Pending • Active • Archived • Deleted
- date-created	string	Date received for academic request, date created for content request

- last-modified	DateTime	Date and time the content item was last modified.
- rejection- message	string	A rejection message given by the DCS user. Values returned from API call shown below (with DCS screen values in brackets): • ExtentExceed (Extent limits exceed CLA Licence limits) • ExcludedFromCLA (Excluded from CLA Licence) • ExtractLimitReached (Extract limit reached for course) • NoPermission (Cannot get permissions to use) • NoSourceContent (Cannot source content) • NotOwned (Content not owned by institution) • Other (Other)
- notes	string	A rejection or published note If the content-status is Rejected, the notes field will show the rejection message entered by the user. Once a link has been generated within the DCS, and the content can be considered as Published, then a Published note can be entered, and this is stored in the Notes field.
- licence	string	The content request licence
- Source	String	Content Sources (ehess, Internal)
- bibliographic-details	object[]	Content item's bibliographic data
-- type	string	Possible outputs: • Book • Journal
-- identifier	string	Content Item's ISN
-- DOI	string	Journal's DOI if the content is a journal
-- title	string	Content's title
-- extract-title	string	Content's article or chapter title
-- publication- form	string	Possible outputs: • Print • Digital
-- year	string	The year the content was published
-- volume	string	The content's volume number
-- issue	string	The content's issue number
-- page-range	string	The page range extracted from the content

-- author	string	The author of the content
-- colour-scale	string	The content's colour scale. Possible outputs: • BlackAndWhite • Greyscale • Colour
-- publisher	string	The content's publisher
-- extract- author	string	The content's extract author
-- chapter- number	string	The content's chapter number
-- edition	string	The content's edition
-- book-pages	int	The content's book page count
-- publication- place	string	The content's publication place
-- OCR	bool	Does the content have optical character recognition?
-- file-size	double	The content's file size in KB
-- subtitle	string	The book subtitle


```
{
  "status": "string",
  "status-code": 0,
  "status-message": "string",
  "total-results": 0,
  "HEI": "string",
  "course-ID": 0,
  "content-items": {
    "content-GUID": "string",
    "content-status": "string",
    "licence": "string",
    "rejection-message": "string",
    "notes": "string",
    "source": "string",
    "date-created": "string",
    "last-modified": "string",
    "content-URL": "string",
    "bibliographic-details": {
      "type": "string",
      "identifier": "string",
      "DOI": "string",
      "title": "string",
      "extract-title": "string",
      "publication-form": "string",
      "year": "string",
      "volume": "string",
      "issue": "string",
      "page-range": "string",
      "author": "string",
      "colour-scale": "string",
      "publisher": "string",
      "extract-author": "string",
      "chapter-number": "string",
      "edition": "string",
      "book-pages": 0,
      "publication-place": "string",
```

```
      "OCR": true,
      "file-size": 0.0,
      "subtitle": "string"
    }
  }
}
```


Error Response


```
{
   "status": "error",
   "status-code": 1,
   "status-message": "Institution not found"
}
{
   "status": "error",
   "status-code": 2,
   "status-message": "Course not found"
}
{
   "status": "error",
   "status-code": 6,
   "status-message": " Invalid Parameters: Course code or request Id number is
   required"
}
```


Sample Call

```
GET https://{host}/v3/GetCourseContent?hei=209&code=ENG101
HTTP/1.1
authorization: Basic YourBase64encodedusername:password
```

Sample Response

GetCourseContent Response


```
{
  "total-results": 1,
  "hei": "CLA",
  "course-ID": 284559,
  "content-items": [{
    "content-GUID": "2998e0f4-db2d-e811-80bf-002590aca7cd",
    "bibliographic-details": {
      "type": "Book",
      "identifier": "9780745321455",
      "DOI": null,
      "title": "State resistance to globalisation in Cuba",
      "extract-title": "The causes and impact of Cuba's crisis in the 1990s",
      "publication-form": "Print",
      "year": "2004",
      "volume": null,
      "issue": null,
      "page-range": "86-151",
      "author": "Carmona Baez, Antonio",
      "colour-scale": "BlackAndWhite",
      "publisher": "Pluto Press",
      "extractAuthor": "Carmona Boez, Antonio",
      "chapter-number": "3",
      "edition": null,
```

```
      "book-pages": 300,
      "publication-place": "London",
      "OCR": false,
      "file-size": 248.05,
      "subtitle": null
    },
    "content-URL": "https://demo-dcs.cla.co.uk/secure/link?id=2998e0f4-db2d-e811-80bf-002590aca7cd",
    "request-id": 261006,
    "previous-year-id": 187698,
    "content-status": "Active",
    "date-created": "10/May/2019 10:21",
    "last-modified": "31/Jul/2019 14:41",
    "rejection-message": "",
    "notes": null,
    "licence": "Covered by CLA Licence",
    "source": "Institution"
  }],
  "status": "ok",
  "status-code": 100,
  "status-message": "Success"
}
```

Submit Request

URL: /SubmitRequest

Method: POST

Authentication: Required

Parameters**:**

Parameter name	Type	Value type	Relevant for	De	Mandatory
hei	Query	integer	Book/ Journal	Institution id to which the academic request will be submitted. User must be an API user subscribed to this institution.	Yes
AcademicName	Query	string	Book/ Journal	Lecturer Name	Yes
EmailAddress	Query	string	Book/ Journal	Lecturer Email	Yes
IsBook	Query	boolean	Book/ Journal	Is book. True/False	Yes
Faculty	Query	string	Book/ Journal	The faculty in which the course is run	No
Department	Query	string	Book/ Journal	The department in which the course is run	No
CourseCode	Query	string	Book/ Journal	Course code of the desired course	No
CourseStartDate	Query	string	Book/ Journal	The start date of the course. Format - date-time (as date-time in RFC3339).	No
ISN	Query	string	Book/ Journal	The International Standard Number of the Manifestation, required if title is missing	Yes, if Title isn't supplied No, if Title supplied
Title	Query	string	Book/ Journal	The book title or journal title, required if ISN is missing	Yes if ISN isn't supplied No otherwise

ExtractTitle	Query	string	Book/ Journal	The chapter title or article title, required for a book if page range not specified	Yes, if IsBook is True and Page Range is blank Otherwise no
Year	Query	string	Book/ Journal	Year of the Publication	No
Issue	Query	string	Journal	Issue of the article	No
Volume	Query	string	Journal	Volume of the article	No
PageRange	Query	string	Book/ Journal	The page range of the extract wanted, not required for a book if extract title is defined	Yes if IsBook = True AND ExtractTitle is not supplied No otherwise
DeliveryDate	Query	string	Book/ Journal	The date the extract is required to be published by. Format - date-time (as date-time in RFC3339).	No
Notes	Query	string	Book/ Journal	Some extra notes to help supply the correct extract	No
Subtitle	Query	string	Book	Subtitle of the book	No
Edition	Query	string	Book	Edition of the ook	No
ChapterNumber	Query	string	Book	Edition of the book	No
ExtractAuthor	Query	string	Book/ Journal	Author of the article or the books chapter	No
Publisher	Query	string	Book/J ournal	Publisher of the book/journal	No
TotalNumberofPages	Query	integer	Book	Number of pages in the book	No
PublicationPlace	Query	string	Book/J ournal	Publication Place of the book/journal	No
Contributor	Query	string	Book	The book's contributor	No
DOI	Query	string	Journal	DOI of the article	No

Response fields

Field	Type	Description
status	string	The status of the API response ("ok", "error")
status-code	integer	The status code of the API response, as detailed in Appendix 2 of this document.
status-message	string	Dependent on whether the request was successful this will either show "Success" or an error message
request-id	integer	The ID of the request

Success Response

```
{
  "request-id": string,
  "status": string,
  "status-code": integer,
  "status-message": string
}
```

Sample Error Responses

```
{
  "status": "error",
  "status-code": 1, "status-message": "Institution not found"
}

{
  "status": "error",
  "status-code": 7, "status-message": “ISBN or Title is mandatory"
}

{
  "status": "error",
  "status-code": 7, "status-message": “Page range or extract title
  is mandatory."
}
```

Sample Call

```
POST
https://{host}/v3/SubmitRequest?hei=195&AcademicName=Mr
test&EmailAddress=test@test.test&IsBook=True&CourseCode=test&Title=Test Book&
PageRange=1-10

authorization: Basic YourBase64encodedusername:password
```

Sample Response


```
{
  "request-id": 428858,
  "status": "ok",
  "status-code": 100,
  "status-message": "Success"
}
```

Submit Course

URL: /SubmitCourse

Method: POST

Authentication: Required

Parameters**:**

Parameter	Parameter Type	Data Type	Description	Mandatory
hei	Query	integer	Institution id where the course will be submitted. User must be a lecturer in this institution or must be an API user subscribed to this institution.	True
CourseCode	Query	string	Course Code	True
CourseName	Query	string	Course Name	True
NumberOfWeeks	Query	integer	Number of Weeks defaults to 52 if left blank	False
NumberOfStudents	Query	integer	The number of students in the course. defaults to 0 if left blank	False
Department	Query	string	The department the course is run in	False
Subject	Query	string	The Subject of the course	False
LeadLecturer	Query	string	The Lead Lecturer, required if LeadLecturerEmail is filled in	False
LeadLecturerEmail	Query	string	TLead Lecturer Email, required if LeadLecturer is filled in	False

Response fields

Field	Type	Description
status	string	The status of the API response ("ok", "error")
status-code	integer	The status code of the API response, as detailed in Appendix 2 of this document.
status-message	string	Dependent on whether the request was successful this will either show "Success" or an error message
request-id	integer	The ID of the request

Success Response

```
{
  "course-Code": string,
  "status": string,
  "status-code": integer,
  "status-message": string
}
```

Error Response

```
{
  "status": "error",
  "status-code": 1, "status-message": "Institution not found"
}
```

```
{
  "status": "error",
  "status-code": 7, "status-message": "Course code already exists"
}
```

```
{
  "status": "error",
  "status-code": 7, "status-message": "Number of weeks must be between 0
                     and 52.

                     Lead Lecturer Name is mandatory if Lead Lecturer
                     Email is filled in.

                     Lead Lecturer Email is not a valid email."
}
```

Sample Call

```
POST https://{host}/v3/SubmitCourse?hei=195&CourseCode=Test1&CourseName=Test
authorization: Basic YourBase64encodedusername:password
```

Sample Response

```
{
  "course-Code": Test1,
  "status": "ok",
  "status-code": 100,
  "status-message": "Success"
}
```

Amend Course

URL: /AmendCourse

Method: PUT

Authentication: Required

Parameters**:**

Parameter name	Type	Value type	Description	Mandatory
hei	Query	integer	Institution id	Yes
CourseCode	Query	string	Code of the Course to be ameded	Yes
NewCourseCode	Query	string	Course Code to be changed to	No
CourseName	Query	string	Course Name to be changed to	No
NumberOfWeeks	Query	integer	Number of Weeks defaults to 52 if left blank	No
NumberOfStudents	Query	integer	The number of students in the course. defaults to 0 if left blank	No
Department	Query	String	The department the course is run in	No
Subject	Query	String	The Subject of the course	No
LeadLecturer	Query	String	The Lead Lecturer, required if LeadLecturerEmail is filled in	No
LeadLecturerEmail	Query	String	The Lead Lecturer Email, required if LeadLecturer is filled in	Yes, if LeadLecturer supplied, otherwise No

Status	Query	String	Course Status: • Active • Archived • Deleted Please note that Delete will delete the course and all content and cannot be undone. Please use with caution.	No

Response fields

Field	Type	Description
status	string	The status of the API response ("ok", "error")
status-code	integer	The status code of the API response, as detailed in Appendix 2 of this document.
status-message	string	Dependent on whether the request was successful this will either show "Success" or an error message
request-id	integer	The ID of the request

Success Response

```
{
  "course-Code": string,
  "status": string,
  "status-code": integer,
  "status-message": string
}
```

Error Response

```
{
        "status": "error",
        "status-code": 1,
        "status-message":
        "Institution not found"
}
```

Sample Call

```
PUT https://{host}/v3/Amendcourse?HEI=195&amp; CourseCode=ACBTest1&amp;
CourseName=ACBTest1&amp; NumberofWeeks=52&amp; NumberofStudents=0&amp;
Department=CLA&amp; Subject=IT&amp; LeadLecturer=Alex Cregan-Bird&amp;
LeadLecturerEmail=alex.cregan-bird@blueyonder.co.uk&amp; Status=Deleted
HTTP/1.1
```

authorization: Basic YourBase64encodedusername:password
Ocp-Apim-Subscription-Key: [your CLA api subscription key]

Sample Response

```
{
  "course-Code": Test1,
  "status": "ok",
  "status-code": 100,
  "status-message": "Success"
}
```

Appendix

Appendix 1 - Error Message

Response fields

Field	Type	Description
status	string	The status of the API response ("ok", "error")
status-code	int	Status-code of the API response
status-message	string	Message related to the API status code

Status Codes

Code	Description
1	Institution not found
2	Course not found
3	Could not authenticate user
4	Internal server error
5	User not subscribed to HEI
6	Invalid parameter
7	Mandatory fields not supplied
100	Success

Appendix 2 - Content Statuses

Status	Description	Available in GetCourseContent call?	Content can be accessed through links by students?
Active	This means the content has been published and is usable by students	Yes	Yes
Pending	Pending content is being prepared by librarians in order to become active	Yes	No
Archived	Archived content is temporarily put into this state so that it can be used at another time without being setup again	Yes	No
Deleted	Deletion occurs when content is not going to be used in the foreseeable future	Yes	No
Rejected	Rejected content has been marked as not being possible for content to be provided. There could be a number of reasons for the librarian to reject a request.	Yes	No

Appendix 3 - Academic Year Timeline

Key dates

- 1st August - new licence year starts.
- June 1st – June 15th - course content rolled over into next academic year.
- Beginning of August - CLA Permissions re-check